

Application Development for Mobile and Ubiquitous Computing

MENSA 2.0

Martin Wudenka, Lucas Vogel

Idea

- Images of the meal (also on iOS)
- Get all available Data (more than what is provided by OpenMensa)
- Allergy filtering
- Favorite meal alert
- high loading Speed + good usability
- Offline check



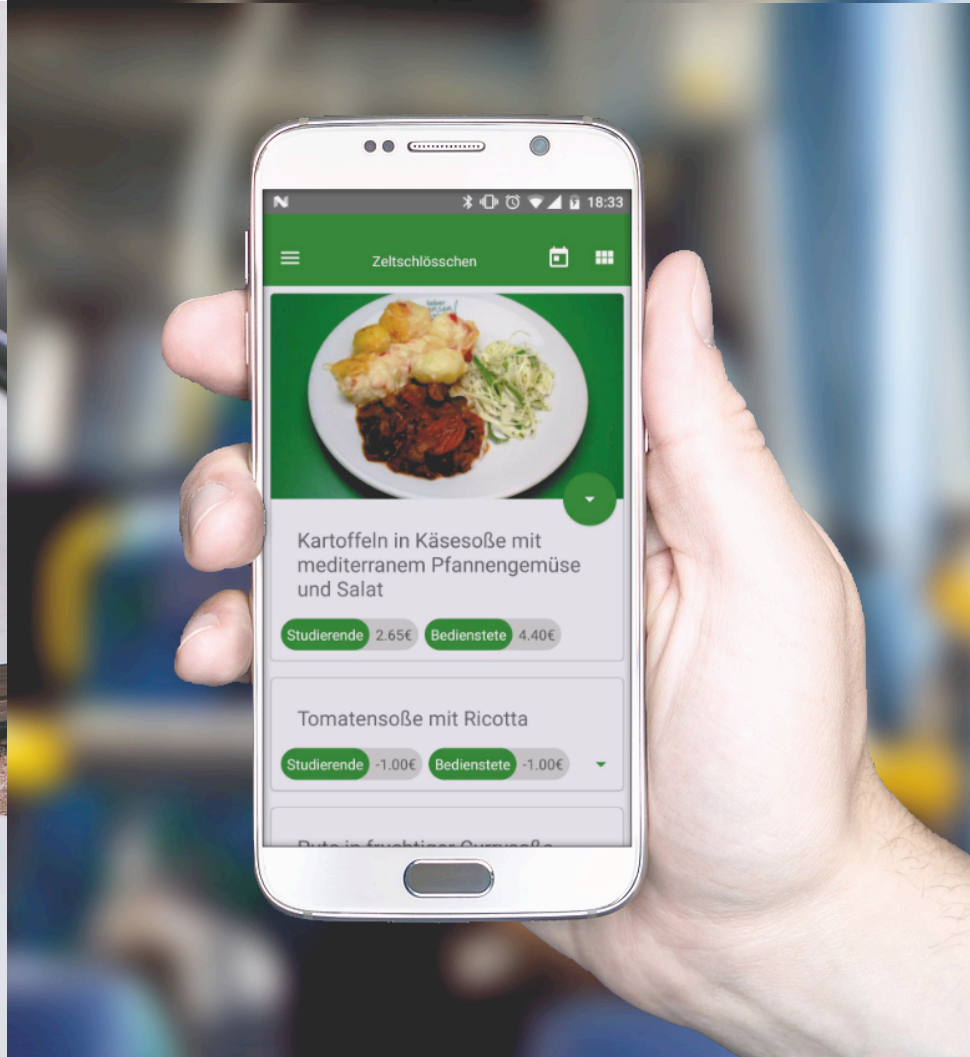
Challenges



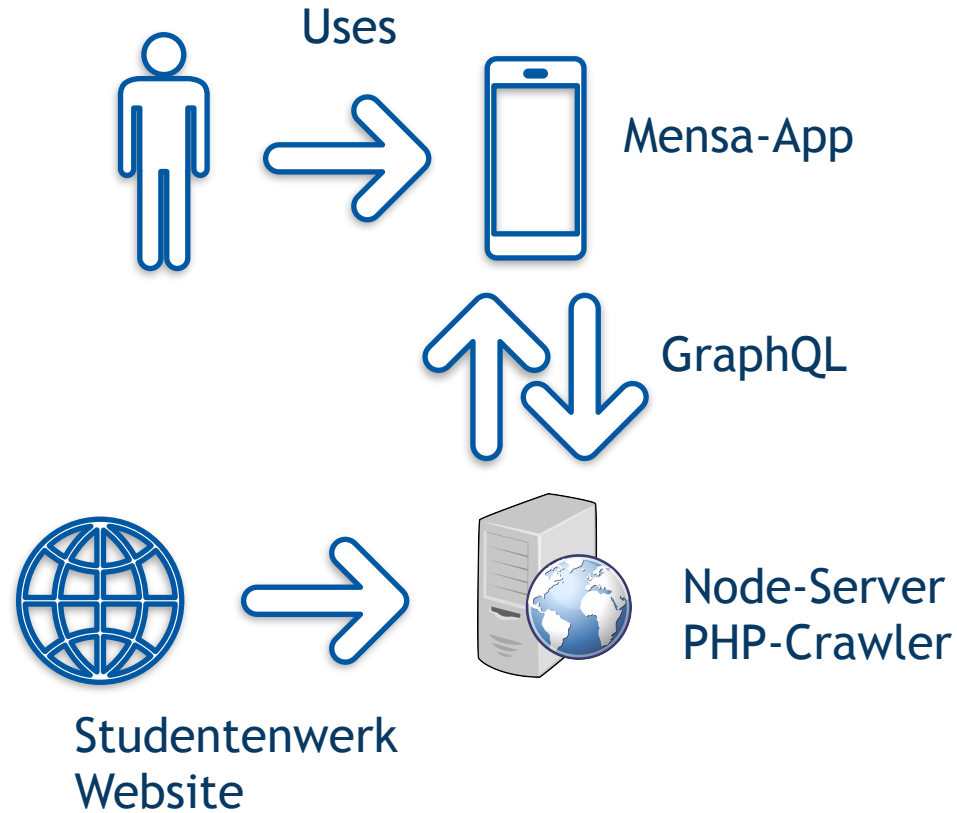
Connectivity &
Offline

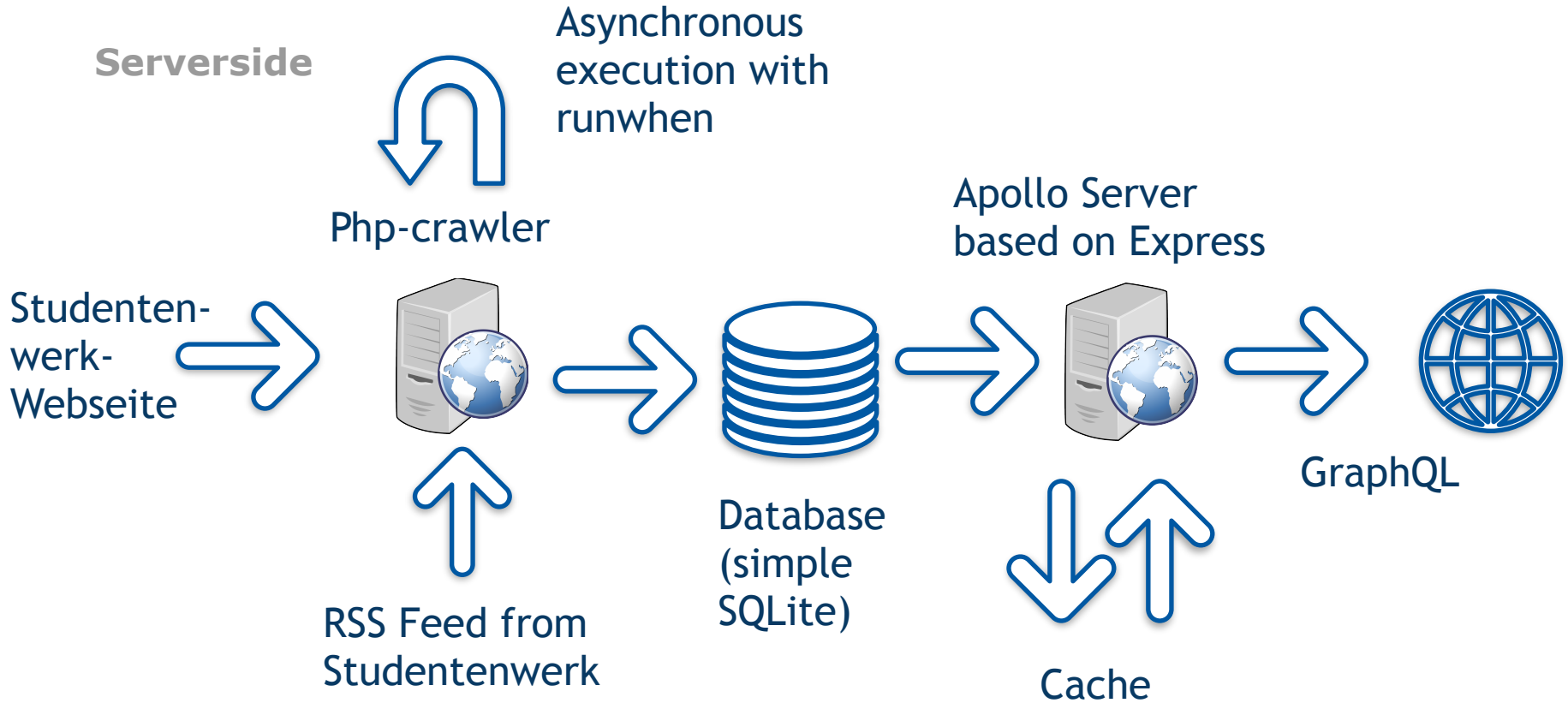


Energy

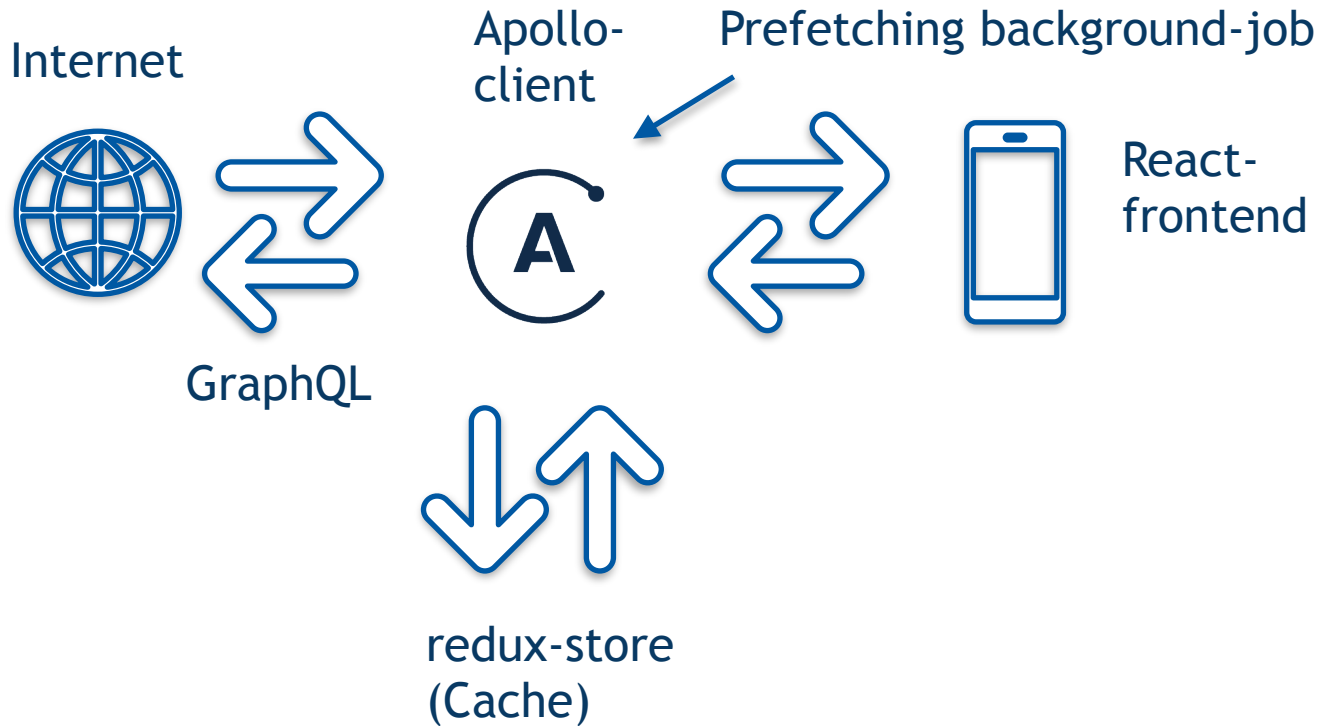


Application Szenario

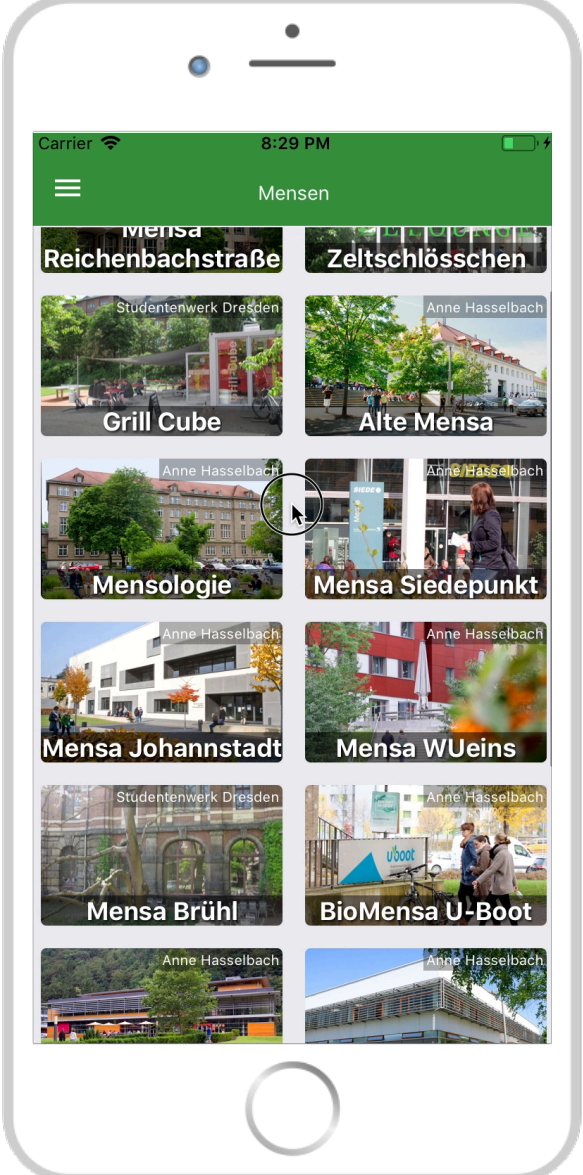
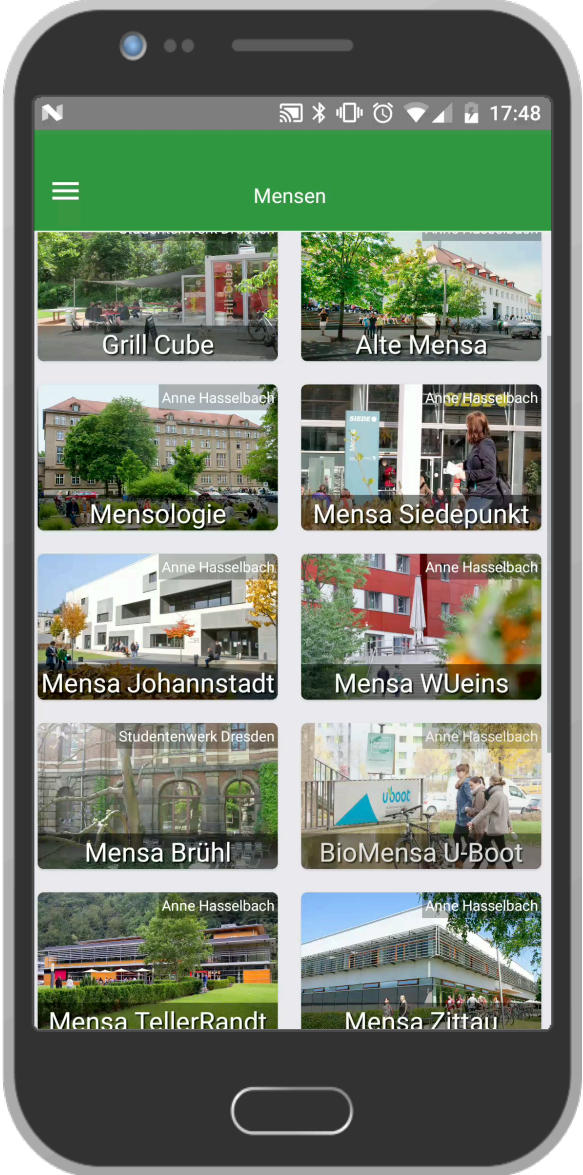




Clientside



Current Status



Connectivity & Offline: Adaptation and Context Information

- Detection via the NetInfo package
- Can differentiate between Wifi, none, cellular and 2g, 3g and 4g
- Based on GraphQL, we can change the amount of data received
- If slow connection exists (slower than 2g), we do not load Pictures (strong data reduction)
- No Background-job on cellular, only Wifi
- Change appearance of Card (next slide)

```

    if (this.props.source !== nextProps.source) {
      this.checkInternetConnection()
    }
  }

  checkInternetConnection = () => {
    NetInfo.getConnectionInfo().then(connectionInfo => {
      let goodConnection = false
      if (connectionInfo.type === 'wifi') {
        goodConnection = true
      } else if (
        connectionInfo.type === 'cellular' &&
        ['3g', '4g'].includes(connectionInfo.effectiveType)
      ) {
        goodConnection = true
      }

      this.setState({
        goodConnection
      })
    })
  }

  render() {
    const { source } = this.props
    const { goodConnection } = this.state
  
```

Context Features

```
componentDidMount() {  
  BackgroundJob.schedule({  
    jobKey: PREFETCHING_JOB,  
    period: 14400000,  
    timeout: 10000,  
    networkType: NETWORK_TYPE_UNMETERED  
  });  
}
```

Background Job

```
BackgroundJob.register({  
  jobKey: PREFETCHING_JOB,  
  job: () =>  
    configureStore({  
      persistCallback: (store, client) => {  
        var format = 'hh:mm:ss';  
        const beforeTime = moment('01:00:00', format);  
        const afterTime = moment('4:59:59', format);  
        if (moment().isBetween(beforeTime, afterTime)) {  
          client.resetStore();  
          client.query({  
            query: prefetchingQuery,  
            variables: { from: moment().format('YYYY-MM-DD') }  
          });  
        }  
      }  
    })  
});
```

Trigger at Night

Adaptation Mechanisms

Cache Lookup

```

export default graphql(MenuQuery, {
  options: props => ({
    variables: {
      mensaId: props.mensa.id,
      day: moment(props.day).format('YYYY-MM-DD')
    },
    fetchPolicy: 'cache-only'
  })
})(MenuImageQualityDecider);
    
```

Image Quality Decider

```

if (
  moment(day).isAfter(moment(), 'day') ||
  (moment(day).isSame(moment(), 'day') &&
    moment().isBefore(moment('11:35:00', 'hh:mm:ss')))
) {
  return <MenuNone mensa={mensa} day={day} />;
}

if (
  (connectionQuality === 'good' &&
    (batteryLevel >= 0.3 || charging)) ||
  cacheQuality === 'high'
) {
  return <MenuHigh mensa={mensa} day={day} />;
}


if (
  (connectionQuality === 'bad' &&
    (batteryLevel >= 0.3 || charging)) ||
  cacheQuality === 'low'
) {
  return <MenuLow mensa={mensa} day={day} />;
}

if (cacheQuality === 'none' && connectionQuality !== 'none') {
  return <MenuNone mensa={mensa} day={day} />;
}
    
```

```

return (
  <View>
    <Text>Keine Internetverbindung</Text>
    <Button
      onPress={data.refetch}
      title="Erneut Probieren!"
    />
  </View>
)
    
```

Adaption based on Connection & Informations



© Studentenwerk Dresden

Ausgabe rechts

Gemüsebällchen mit Chili-Lauchsoße, dazu glasierte Honigmöhren und Kokosreis

Studierende 2,65€ Bedienstete 4,35€

Online & Full Data

Abendangebot

Abendangebot:
Bratkartoffelpfanne mit Kassler
und Gemüse oder Bratkartoffeln
mit Tomate und Mozzarella, dazu
bunter Chinakohlsalat

Studierende 2,65€

Bedienstete 4,35€

Offline (cached Data)
or no more available Data

Energy: Adaptation and Context Information

- using the react-native-battery-status package
- detect battery status in percent
- If battery is low, pause background-jobs
 - do not load meal-images
 - Adaptation like before
 - only take data from cache



Open Issues and Lessons Learned

- Better implementation of features
- better case-handling from backend

- Getting good performance with React is possible, but hard
- The Studentenwerk-Website is not crawler-friendly
- GraphQL is really nice, but just in JavaScript



Open Issues and Lessons Learned

- 3rd party packages are sometimes unstable ->
 - had to remove tabs
- not enough time due to other projects to implement allergy filtering and favorite meal
- cross-platform development is tedious, at least in terms of layout

Upgrade react-native-tab-view #3064

Merged
 kelset merged 2 commits into `master` from `@satya164/upgrade-tab-view` 19 days ago

Conversation 5
Commits 2
Files changed 7



satya164 commented on 2 Dec 2017 • edited ▾

Collaborator + 🧑

With this update, the `lazy` option has been removed, so it's a breaking change.

- It didn't work great with more than 2 tabs
- It's not very performant to listen to all events during the gestures since we're trying to move away towards native gestures.
- It basically triggered a re-render of the whole tab view mid-gesture, which could cause jank

I understand that it's still desirable to lazily load tabs, so I think it's time we implement `focus events` so that this can be implemented in userland. The approach I'd go with is this:

- Implement a placeholder component for each tab screen based on a `loaded` state
- When a tab screen is focused, toggle the `loaded` state and replace placeholder component with actual content

👍 6

👎 25

😬 9