AAD / Backend Development

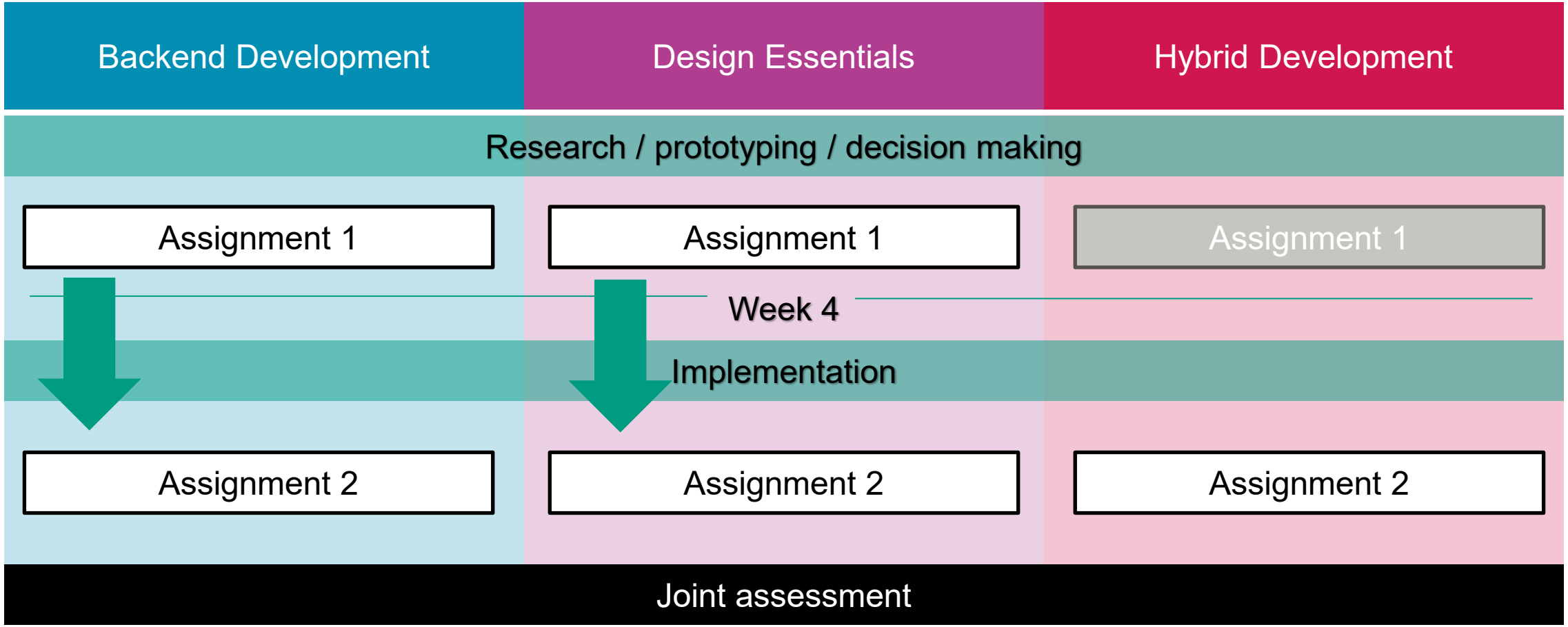
API flavors

Erik van der Arend / Jan Willem Boer

23/24 Q3

SAXION
UNIVERSITY OF
APPLIED SCIENCES

AAD and Backend Development

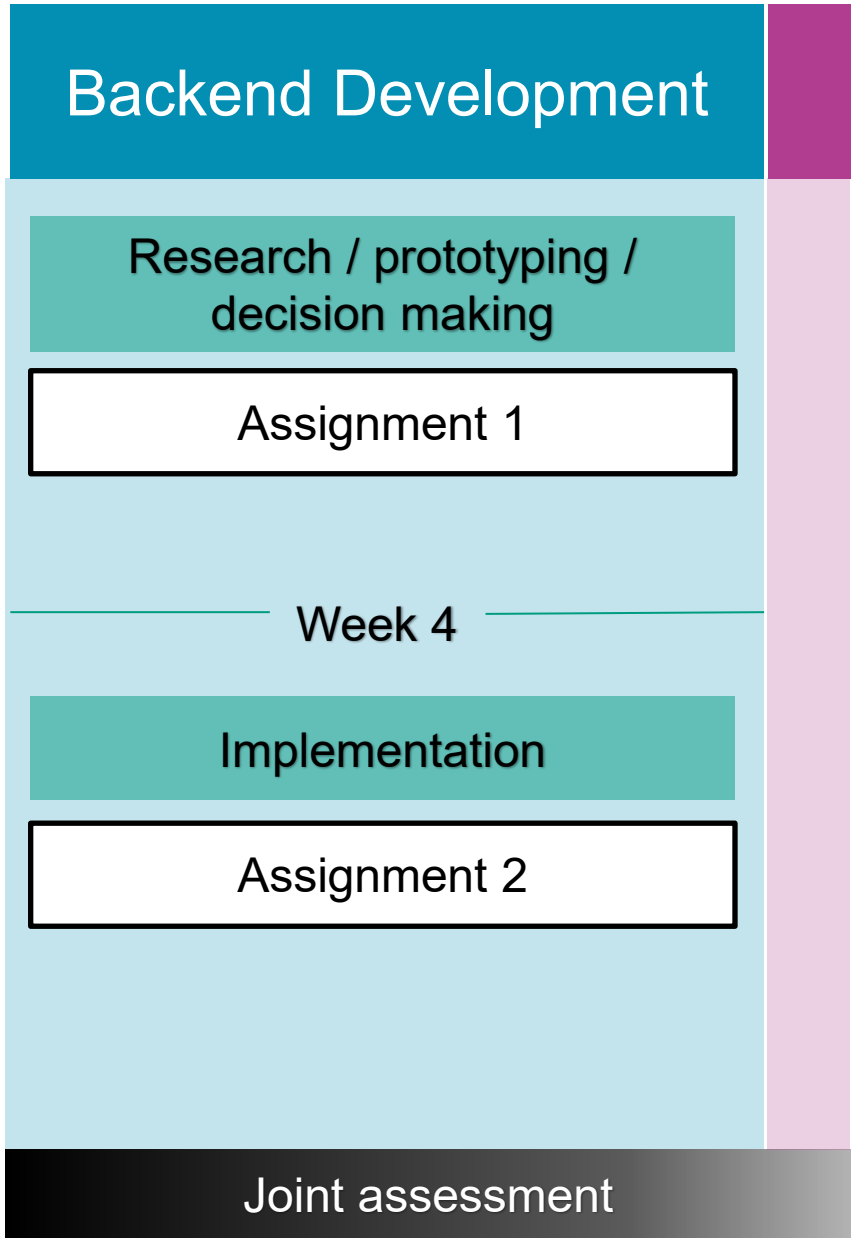


BD assignments

Assignment 1
- Select an *advanced track (AT)*
- Find an idea for your application
- Research backend techniques that fit the idea and AT

Official deadline is week 9; but when submitted at start week 4, you get feedback and can resubmit at week 9.

Assignment 2
- Create the backend application, including advanced track
- Create technical documentation
- Dockerize the application
- Create a show-and-tell document about your app and AT
- Do a short presentation about your AT in class



BD assignment 1 – research, prototyping, decision making

Phase 0
- Create a structured list of criteria for the three subjects: API, framework and DB.
- Create a long list of frameworks / techniques

Phase 1
- For each subject, compare at least three frameworks / techniques
- Choose two combinations based on the comparison

Phase 2
- Implement a prototype of your application in the two combinations

Phase 3
- Document it and draw a conclusion for the combination of techniques for assignment 2

BD assignment 1 – rubric

Prerequisites for grading:

- Phases 1-3 covered in the document
- Two working prototypes
- The prototypes have the basic requirements (next slide)
- A large part of the prototypes is for researching the chosen advanced track.
- No build artifacts in the submission.

BD assignment 1 – rubric

Basic application requirements for assignment 1:

- Has a database to persist data
- Has a formal API
- At least three entities/resources having CRUD functionality
- One of them should have a one-to-many relation with another entity/resource
- The code is layered
- There are instructions how to test the application

See assignment 1 for the details.

BD assignment 1 – rubric

Framework selection: 20 points (= 2 actual points)

- Description for each **framework/technique** that you looked at
- Description of all **criteria + weight**, with reasoning
- Extra care for criteria for the **advanced track**
- Use a **systematic way** to score each framework/technique.
- **Justify** decisions, arguments by references
- Create an **overview** of the final scores (table)
  with conclusion about which two frameworks you will use for the prototype

BD assignment 1 – rubric

Prototyping: 20 points (= 2 actual points)

- What did you find doing the prototypes?
  Keep track of this while you create them
- Use the same criteria that you used earlier as guideline to structure both reports
- With, of course, extra care for criteria for the advanced track
- Draw a final, logical conclusion.

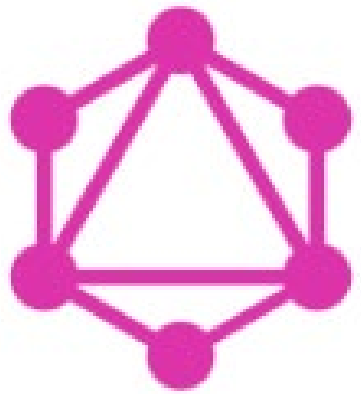
API techniques

GraphQL

- Schema-based
- Text over HTTP (like REST)
- Client(s) use a query language to specify what they need
- Cross-platform

GraphQL: APIs for humans

GraphQL is the developer-friendly query language for the modern web. It transforms how apps fetch data from an API, enabling you to **get exactly what you need with a single query**—instead of wrangling responses from a patchwork of REST endpoints.



GraphQL schema example

```
type Query {
    recipe(id: ID): Recipe
    pid: Int
}
type Recipe {
    id: ID!
    name: String!
    steps: String
    ingredients: [Ingredient]!
}
type Ingredient {
    id: ID!
    name: String!
    quantity: String
}
```

GraphQL query example

```
type Query {
    recipe(id: ID): Recipe
    pid: Int
}
type Recipe {
    id: ID!
    name: String!
    steps: String
    ingredients: [Ingredient]!
}
type Ingredient {
    id: ID!
    name: String!
    quantity: String
}
```

```
{
  recipe(id:1) {
    id,
    name
  }
}
```

```
{
  recipe(id:1) {
    id,
    name,
    ingredients {
      id,
      name,
      quantity
    }
  }
}
```

GraphQL resolvers

```
type Query {
    recipe(id: ID): Recipe
    pid: Int
}
type Recipe {
    id: ID!
    name: String!
    steps: String
    ingredients: [Ingredient]!
}
type Ingredient {
    id: ID!
    name: String!
    quantity: String
}
```

```
const resolvers = {
    Query: {
        pid: () => process.pid,
        recipe: async (_obj, {id}) => data.getRecipe(id)
    },
    Recipe: {
        ingredients: async(obj) => data.getIngredients(obj.id)
    }
};
```

```
function getRecipe(id) {
    if (id != 1) {
        throw new Error(`only recipe 1 is supported`);
    }
    return {
        id,
        name: "Boerenkoolstamppot",
        steps: "Bake bacon, onion and garlic until brown. Lower fire
        some cooking water to the mesh. Throw in salt and pepper."
    }
}
```

Practice

- Clone or fork the project at

https://github.com/janwillemboer/graphql-project
or
https://edu.nl/wgcqv

Start the server and go to localhost:3000/graphiql

- Try out the queries from the previous slides to see if the server works.
- Can you get the recipe steps in the response?
- Create a nodejs client that logs the recipe to the console.

SAXION UNIVERSITY OF APPLIED SCIENCES

gRPC

- Schema-based
- Binary protocol over HTTP
- Cross-platform
- Clients use API like local methods

gRPC schema example

```
service RecipeService {
    rpc GetRecipe(RecipeRequest) returns (Recipe) {}
    rpc GetMetaData(Empty) returns (Meta) {}
}

message RecipeRequest {
    int32 id = 1;
}
message Meta {
    int32 pid = 2;
}
message Empty {}

message Recipe {
    int32 id = 1;
    string name = 2;
    string steps = 3;
    repeated Ingredient ingredients = 4;
}

message Ingredient {
    int32 id = 1;
    string name = 2;
    string quantity = 3;
}
```

Practice

- Clone or fork the project at

https://github.com/janwillemboer/grpc-project

or

https://edu.nl/phaae

- Start the server → node index.js
- Finish the client to fetch the recipe from the server (client.js)
  For reference, use

https://grpc.io/docs/languages/node/quickstart/

SAXION UNIVERSITY OF APPLIED SCIENCES

NOW WHAT

Did you already kickstart your assignment 1?

Discuss your setup with the teacher today.

Reminder for the steps:

- Find an **app idea**
- Select an **advanced track** from the document on Blackboard that fits the idea
- Initialize a **repository** @ https://repo.hboictlab.nl/ under 4.12 AAD / BD.
- Subscribe for a **group on blackboard**
- Read about the details of **BD assignment 1** on Blackboard and continue working on it.