Implementation 2

Assessment 2 Team

Team 12

Richard Liiv

Umer Fakher

William Walton

James Frost

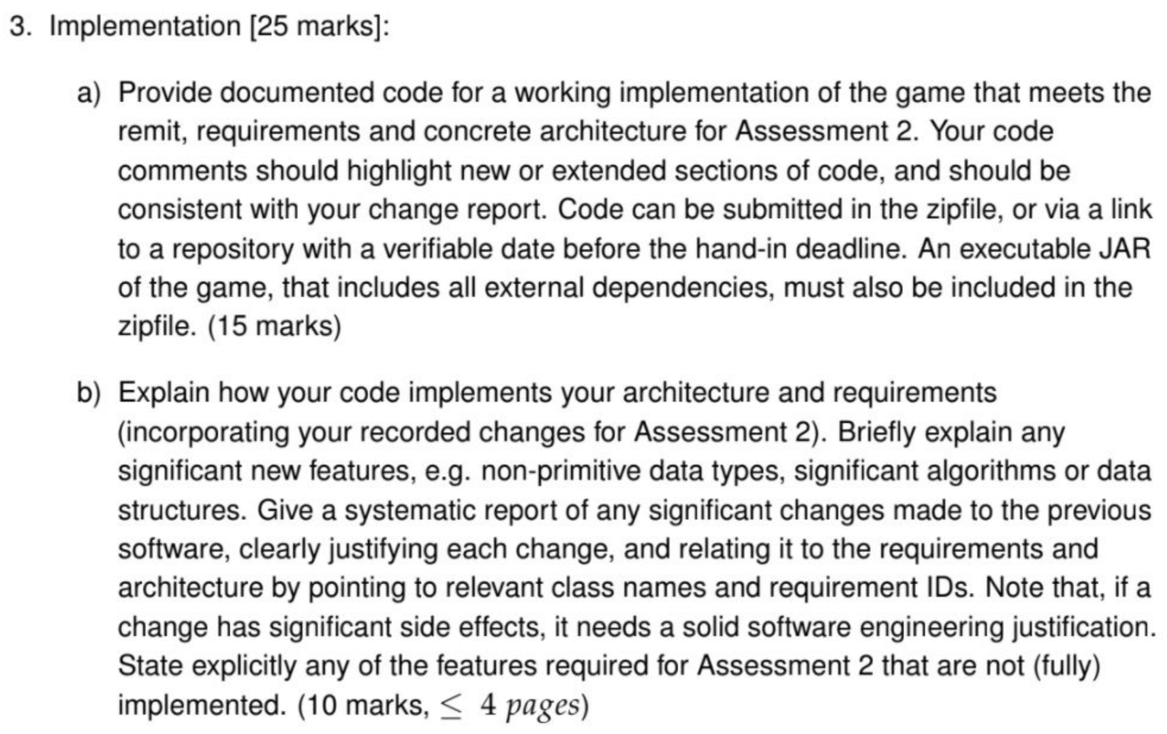
Olly Wortley

Joe Cambridge

b)

DELETE THIS page

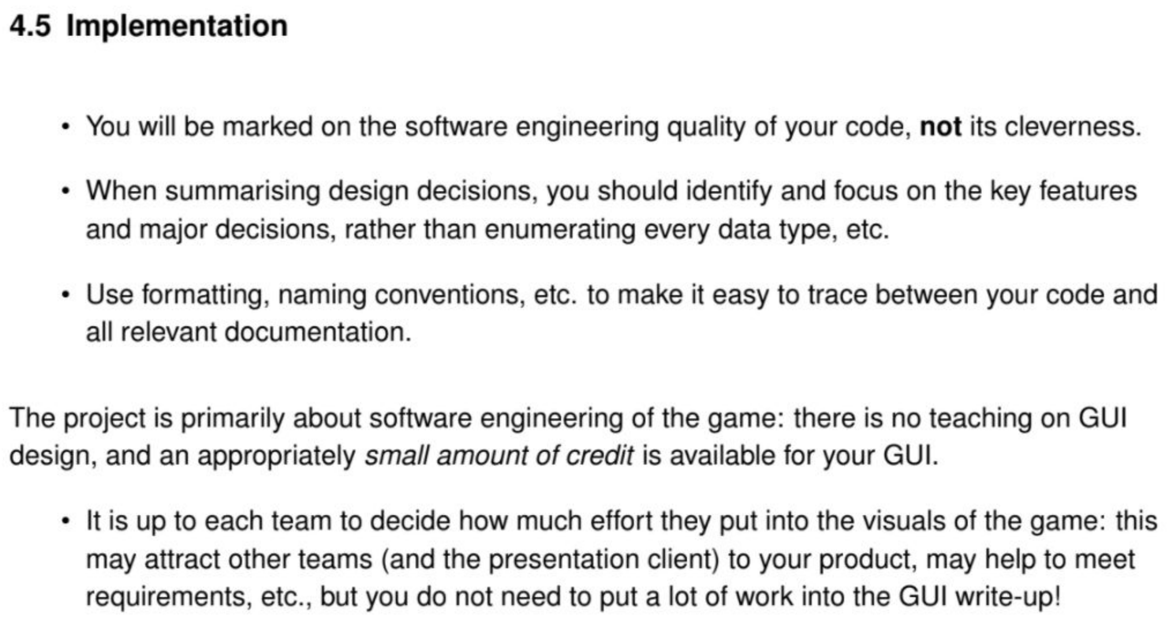
Page limit 4



3. Implementation [25 marks]:

   a) Provide documented code for a working implementation of the game that meets the remit, requirements and concrete architecture for Assessment 2. Your code comments should highlight new or extended sections of code, and should be consistent with your change report. Code can be submitted in the zipfile, or via a link to a repository with a verifiable date before the hand-in deadline. An executable JAR of the game, that includes all external dependencies, must also be included in the zipfile. (15 marks)

   b) Explain how your code implements your architecture and requirements (incorporating your recorded changes for Assessment 2). Briefly explain any significant new features, e.g. non-primitive data types, significant algorithms or data structures. Give a systematic report of any significant changes made to the previous software, clearly justifying each change, and relating it to the requirements and architecture by pointing to relevant class names and requirement IDs. Note that, if a change has significant side effects, it needs a solid software engineering justification. State explicitly any of the features required for Assessment 2 that are not (fully) implemented. (10 marks, $\leq\ 4\ pages$)



4.5 Implementation

- You will be marked on the software engineering quality of your code, **not** its cleverness.
- When summarising design decisions, you should identify and focus on the key features and major decisions, rather than enumerating every data type, etc.
- Use formatting, naming conventions, etc. to make it easy to trace between your code and all relevant documentation.

The project is primarily about software engineering of the game: there is no teaching on GUI design, and an appropriately *small amount of credit* is available for your GUI.

- It is up to each team to decide how much effort they put into the visuals of the game: this may attract other teams (and the presentation client) to your product, may help to meet requirements, etc., but you do not need to put a lot of work into the GUI write-up!

a) JUST A HAND IN

b)

i) Requirements

The requirements for assessment 1 for Team 15 were mostly not changed as they apply to assessment 2. However, changes were made to the requirements tables for the additional assessment 2 requirements. There were 3 new requirements for assessment 2: levels of difficulty, saving and resuming the game and implementing 5 powerups to boost attributes of the boat.

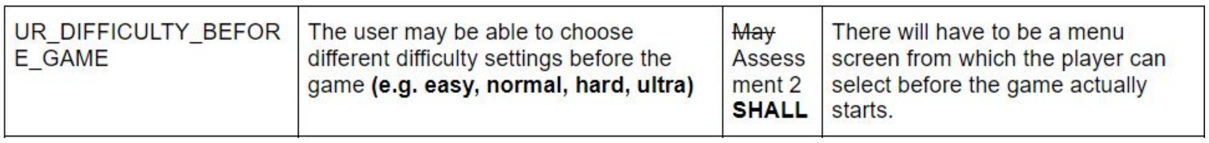
Team 15 had 4 requirements listed in assessment 1's not fully implemented section:

- UR_DIFFICULTY_BEFORE_GAME,
- UR_POWERUPS,
- FR_POWERUPS_RATE,
- NFR_ATTRIBUTES

One not mentioned was FR_DIFFICULTY_SELECTION although we can imply that this is not completed from its parent UR_DIFFICULTY_BEFORE_GAME.

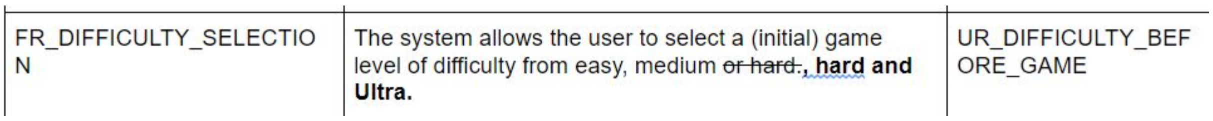
They were very similar to the assessment 2 requirements so we combined them with the new assessment 2 requirements. See the following:

Difficulty



UR_DIFFICULTY_BEFORE_GAME	The user may be able to choose different difficulty settings before the game **(e.g. easy, normal, hard, ultra)**	~~May~~ Assessment 2 **SHALL**	There will have to be a menu screen from which the player can select before the game actually starts.

- UR_DIFFICULTY_BEFORE_GAME (User Requirement)
  - Changed description to clearly define difficulties (easy, medium, hard & ultra) inline with the new requirements for assessment 2.
  - Increased priority to shall for assessment 2



FR_DIFFICULTY_SELECTION	The system allows the user to select a (initial) game level of difficulty from easy, medium ~~or hard.~~, **hard and Ultra.**	UR_DIFFICULTY_BEFORE_GAME

- FR_DIFFICULTY_SELECTION (Functional Requirement)
  - Changed description to clearly define difficulties (easy, medium, hard & ultra) inline with the new requirements for assessment 2.
  - Increased priority to shall for assessment 2

Power Ups

UR_POWERUPS	The user may be able to pick up ~~powerups.~~ **five power-up packs, which can be found floating down the river and be picked up by boats to improve characteristics: health, agility, speed, stamina or a 5th one which improves all at once.**	~~May~~ Assessment 2 **SHALL**	~~We may not have enough time to implement this, due to time constraints. Associated with R8.~~

- UR_POWERUPS Changed (User Requirement)
  - Changed description to clearly define details of the power up packs that we have been assigned inline with the new requirements for assessment 2.
  - Increased priority to shall for assessment 2
  - Removed notes

FR_POWERUP_RATE	The system must decide on an appropriate amount of power ups to spawn during a race.	UR_POWERUPS

- FR_POWERUP_RATE (Functional Requirement)
  - This requirement is linked to the previous one and required no change even though the description of UR_POWERUPS was edited. It is relevant to assessment 2 so is included for context for UR_POWERUPS.
  - It was also not implemented in assessment 1.

  - **Our game does decide on an appropriate amount of power ups to spawn during a race.**

Attributes

NFR_ATTRIBUTES	The system must explain what the different attributes are and how they affect your boat. A **menu screen for boat selection should display the different attributes of each boat type.**	UR_BOAT_UNIQNESS	Before the game starts.

- NFR_ATTRIBUTES
  - This was an incomplete requirement from assessment 1 so will be followed through in assessment 2.
  - Changed description to clearly define details of the power up packs that we have been assigned inline with the new requirements for assessment 2.

    https://github.com/UmerFakher/ENG1Project/pull/7

    Pull request links will help explain and justify this stuff

Save & Resume Game

UR_SAVE_RESUME_GAME	**The players should be allowed to save the state of the game and resume a saved game later**	Assessment 2 **SHALL**	**When the player saves by pausing the game it will save progress made up to the start of the last leg.**

- UR_SAVE_RESUME_GAME
  - A new requirement about saving the game state was added, including the

description, Shall priority and notes inline with the new assessment 2.

Changes made to program:

https://github.com/JoeWrieden/ENG1Project/compare/master...UmerFakher:master

Use for Impl2

Added `protected int difficulty = 0;` to the main game class to hold difficulty with getter and setter (DragonBoatRace.java):

```
public int getDifficulty() {
    return difficulty;
}

public void setDifficulty(int difficulty) {
    this.difficulty = difficulty;
}
```

Added new class `public class DifficultySelectScreen implements Screen`

to let the user select a difficulty

Made more obstacles spawn on higher difficulties (lane.java)

```
int difficulty_mod = 0;

switch (difficulty){
    case 0: difficulty_mod = 0; break;
    case 1: difficulty_mod = 2; break;
    case 2: difficulty_mod = 10; break;
    case 3: difficulty_mod = 30; break;
}

for (int i = 0; i < (11 - Settings.PLAYER_COUNT + round - 1 +
difficulty_mod); i++) {
    replaceObstacle();
}
```

Made the initial 3 obstacle types (the actual obstacles) the most likely to spawn (lane.java)

```
double randD = (ThreadLocalRandom.current().nextGaussian()*3+1); // random
variable normally distributed with mean=1,sd=3
int rand = Math.min(Math.max((int)randD, 0),
ObstacleType.values().length-1); // constrain value between 0 and max
obstacleType index
return new Obstacle(ObstacleType.values()[rand], this.area.getX(),
this.area.getWidth());
```

Added new `ObstacleType` options to include power-ups (obstacletype,java):

```
PU_HEALTH("health_power_up.png", 75, -20, 0, 0, 0),
PU_STAMINA("stamina_power_up.png", 75, 0, 20, 0, 0),
PU_AGILITY("agility_power_up.png", 75, 0, 0, 1, 0),
PU_SPEED("speed_power_up.png", 75, 0, 0, 0, 20),
PU_ALL("all_power_up.png", 75, -20, 20, 1, 20);
```

Added more attributes to obstacles to allow for positive effects (obstacletype,java):

```
private final float staminaMod;
private final float agilityMod;
private final float speedMod;

public float getStaminaMod() {
    return staminaMod;
}

public float getAgilityMod() {
    return agilityMod;
}

public float getSpeedMod() {
    return speedMod;
}
```

Changed main Race game loop to exit on Esc (race.java)

```
if (Gdx.input.isKeyJustPressed(Input.Keys.ESCAPE)){
    getLeaderBoard(game, false);
}
```

Added saving on Esc when in between rounds (roundsscreen.java)

```
if (Gdx.input.isKeyJustPressed(Input.Keys.ESCAPE)) {
    saveToFile("savefile.txt", playerBoat.getBoatType(),
this.game.getPlayerTotalTime(), this.game.getRound(),
this.game.getDifficulty());

    //reset the rounds
    this.game.setRound(1);
    this.game.setScreen(new MainMenuScreen(this.game));
}
```

Added saving to file (roundsscreen.java)

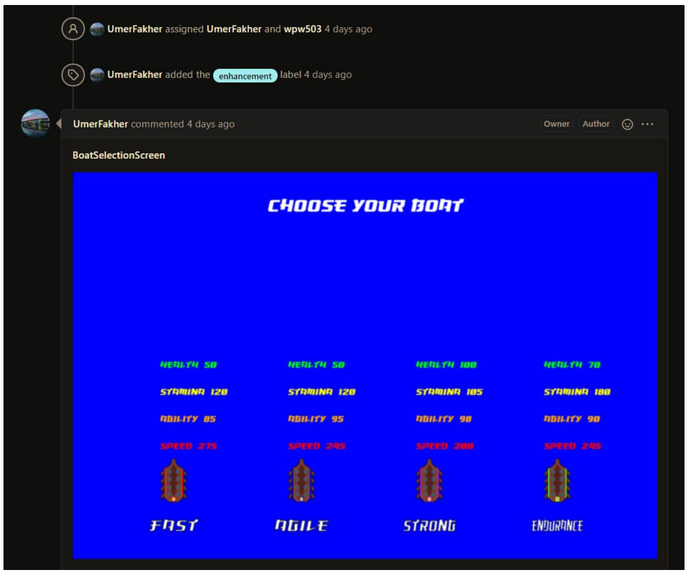
```
public static void saveToFile(String filename, BoatType boatType, float
totalTime, int round, int difficulty) {
    File oldFile = new File(filename);

    try {
        File newFile = new File(filename);
        newFile.createNewFile();
    } catch (IOException e) {
        System.out.println("An error occurred.");
        e.printStackTrace();
    }

    try {
        FileWriter myWriter = new FileWriter(filename, false);
        myWriter.write(boatType.getSaveString()
                + Float.toString(totalTime) + "\n"
                + Integer.toString(round) + "\n"
                + Integer.toString(difficulty) + "\n");
        myWriter.close();
    } catch (IOException e) {
        System.out.println("An error occurred.");
        e.printStackTrace();
    }
}
```

Added load button in main menu with functionality to load from file (maingamescreen.java)

```
loadButton.render(this.game.getBatch());
if (this.loadButton.isHovering() && Gdx.input.isTouched()) {
    List<String> saveData = new ArrayList<>();
    BoatType boat;

    try {
        File myObj = new File("savefile.txt");
        Scanner myReader = new Scanner(myObj);
        while (myReader.hasNextLine()) {
            String data = myReader.nextLine();
            //System.out.println(data);
            saveData.add(data);
        }
        myReader.close();
    } catch (FileNotFoundException e) {
        System.out.println("An error occurred.");
        e.printStackTrace();
    }

    switch (Integer.parseInt(saveData.get(0))){
        case 0: boat = BoatType.FAST; break;
        case 1: boat = BoatType.AGILE; break;
        case 2: boat = BoatType.ENDURANCE; break;
        case 3: boat = BoatType.STRONG; break;
        default: boat = BoatType.FAST; break;
    }

    game.setPlayerTotalTime(Float.parseFloat(saveData.get(1)));
    //minus one needed to offset auto increment happening before the save
    game.setRound(Integer.parseInt(saveData.get(2))-1);
    game.setDifficulty(Integer.parseInt(saveData.get(3)));

    game.setScreen(new MainGameScreen(this.game, boat));
}
```

Added Attributes shown for each boat

https://github.com/UmerFakher/ENG1Project/pull/7



UmerFakher assigned **UmerFakher** and **wpw503** 4 days ago

UmerFakher added the enhancement label 4 days ago

UmerFakher commented 4 days ago

BoatSelectionScreen

I'll add more here - UF