Our Discord Bot is quickly and easily implemented. In order to deploy it correctly on a Discord Server of your own, follow the hereby described steps:

***Implementation:***

1. Open the Bot folder in a source-code editor of your choice (we recommend Visual Studio Code, as we used it for our project) and click on the config.json file.
2. Create / Use a Discord Server that already exists or you are an admin on. In case a Discord Server is already set up, skip to step 3. If there is the need to create a Discord Server, Download the Discord Application. On the left-hand bar, click on the “Add Server” Icon and create a Server.
3. Once the Server is readily available, log into the developer portal of Discord on a Browser of your choice through the following link: <https://discord.com/developers/applications>
4. There, click on the “New Application” Button, type in a name of your likings for the Bot and click “Create”.
5. On the left “Settings” bar, click onto the “Bot” section. Here, under the “Username” field, there is a “Token”. In case an old Token has already been generated and it can not be seen what the old Token is (as this is private), click on “Reset Token” and copy it. If that is not the case, simply copy the Token.
6. Head back into the config.json file and copy this token between the apostrophes like this: “token” : “Enter token here”. This counts for all Tokens/IDs that follow.
7. Head back to the Discord Developer Portal. Here, on the left-hand bar, choose “OAuth2”. After clicking on “Add redirect”, a drop-down menu will open on the left. Select “URL Generator”. Under “Scopes”, select “bot”. A new menu will open up with “Bot permissions”, here simply tick the box “Administrator” and copy the Generated URL below.
8. After generating the Bot’s URL, paste it into a new Tab on your Browser and select the Server you want the Bot to run on. After confirming that you want the Bot to be created on this Server, the Bot should now be visible within the Servers Member-list.
9. Heading back to the Developer Portal, under the “OAuth2” selection, click on “General”. The Client ID should be obtainable here without any further problems, simply copy and paste it into the config.json file under “ClientID”.
10. To obtain the “GuildID” and “musicChannelID”, head into the browser, open a new Discord Tab and click on your freshly created server. Here, select the channel you want our implemented music-feature to be played in. The IDs will be shown in the Websites URL in the following format:  
    https://discord.com/channels/GuildID/musicChannelID
11. Copy these two IDs and paste them into the config.json file. After saving the newly applied changes, the Bot is ready to be deployed.
12. To deploy the bot, within Visual Studio Code (or the source-code editor of your choice), open a new Terminal. To do this on Visual Studio code, simply click the “Terminal” Button on the top bar and create a “New Terminal”.
13. In the Terminal, firstly paste in the command “npm i” to install all packages needed in order for the Bot to function properly. Once the Terminal confirms the completion of installed packages, type “node deploy-commands.js” into that same Terminal. Now, to start the Bot, simply enter one last command into the Terminal: “node index.js”.
14. If all steps were done properly, a message should appear in the Terminal, saying “Ready! Logged in as >your bots name<”. Now, the Bot should be appearing as Online in your server and ready to be used.

***Commands:***

Everything written within these >brackets< is information the User is supposed to fill in.  
Besides the built-in commands from Discord, we have implemented the following:

/ping to check whether the Bot is replying to commands correctly.

/server to check the Server name & member count.

/user which provides information about the user executing this command

/weather >city/country< to fetch weather information for any city or country.

/watch >YouTube URL< to create a Watch2Gether Lobby for you and your friends.

/update >YouTube URL< to add a new Video to the Playlist of the created Watch2Gether Lobby.

/invite to create an invitation link for your friends in order to join your server.

/joke which tells a random joke

/news-locations to firstly get a list of countries where a User can fetch news from,

Combined with /news <location code> to fetch the news from that location.

/play >YouTube URL< which plays your YouTubes videos Audio in the channel you provided.

/rps >Rock, Paper or Scissors< in order to play Rock-Paper-Scissors against the Bot.

/dice which rolls a dice for you.

***New implementations:***

We made the Bot very easy to scale in case a User wants to add new Commands him/herself due to the nature of the SlashCommandBuilder. Most API calls will be following the same implementation as our (example given:) watch.js or weather.js files. If new commands, changes to our commands or any other implementations have been done, before deploying the Bot with the “node index.js” command in the Terminal, the command “node deploy-commands.js” should be executed to ensure that all changes have been registered by the Bot.