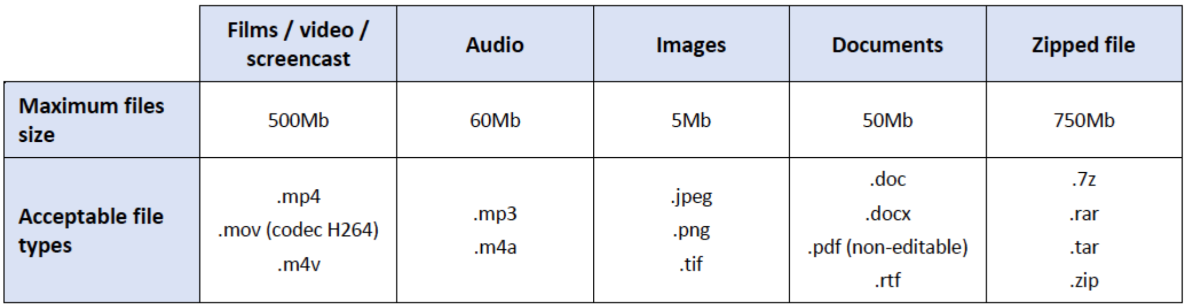
CRITERION D – Functionality and extensibility of the product

→ **Screencast (Video)** showing how your system (solution/product) works; think user documentation.

- Show how your system validates inputs and use every (menu) option
- Use your *test plan* (from criterion B) to fully test your IA solution/product;
- 5 to 7 minutes long, ***do not exceed 7 minutes!***
- For macOS, use QuickTime's *File → Screen recording* option;
- For Windows, you may want to use:
  - ShareX https://getsharex.com/
  - OBS Studio https://obsproject.com/ (also available for macOS, Linux)
  - Free Cam https://www.freescreenrecording.com/
  - Camtasia or SnagIt trials https://www.techsmith.com/
  - Screencast-O-Matic https://screencast-o-matic.com/screen-recorder
  - Or any similar tool you prefer, as long as it outputs a standard video file format with acceptable size (**< 500 MB**), video and sound quality. Please refer to the table below, from the IBO:



	Films / video / screencast	Audio	Images	Documents	Zipped file
Maximum files size	500Mb	60Mb	5Mb	50Mb	750Mb
Acceptable file types	.mp4 .mov (codec H264) .m4v	.mp3 .m4a	.jpeg .png .tif	.doc .docx .pdf (non-editable) .rtf	.7z .rar .tar .zip

→ **Modification and expansion of the system** (post-completion of the development stage)

Suggestions:

- Add your final UML class diagram (made with ObjectAid for Eclipse, EasyUML for Netbeans, or a similar plugin of functionality of your advanced IDE) to the end of the document for criterion B (Solution Overview/Design)
- Explain any modifications you made to your original class design (describe the modifications and justify each of them)
- Create a table including the most important classes in your system and their responsibilities / purposes; include comments on how these classes may be modified by future developers in order to keep up with increasing demands or changes (see example on next page)

Class	Responsibility	Modification/expansion
User	Defines the attributes and methods related to each family member who is given a budget and is allowed to have expenses.	The expenses array may be replaced by an *ArrayList<Expense> expenses = new ArrayList<>();* in the User class, after importing java.util.ArrayList. Class constructors and toString methods in *User* will need to be modified accordingly, plus any array references in the *UserManager* class: add, remove, edit expense, plus save and load methods.
Budget	Defines the attributes and methods related to the monthly budget of each user.	The *calculateGST* method uses GST, a class constant (final), type double declared and initialised on line 3 of the Budget class to 0.07, the prevailing Goods and Services Tax where the client resides and at the time when this IA was developed. Modify this constant and re-compile the code if there are changes in the GST.
…	…	…